Journal on Interactive Systems, 2026, 17:1
ISSN: 2763-7719 • doi: 10.5753/jis.2026.6127

Research Paper

AShE – A Shadow Estimator for Augmented Reality Systems on Mobile Platforms

André Luís Braga Dutra [Federal University of Juiz de Fora | *andre.luis@estudante.ufjf.br*]
Rodrigo Luis de Souza da Silva [Federal University of Juiz de Fora | *rodrigoluis@ufjf.br*]

Institute of Exact Sciences, Computer Science Department, Federal University of Juiz de Fora, Rua José Lourenço Kelmer, s/n - Campus Universitário, São Pedro, Juiz de Fora, MG, 36036-900, Brazil.

Abstract. This work presents a geometric method for estimating the directional light vector in a scene, enabling realistic shadow generation in augmented reality through image segmentation and inverse rendering. The solution is designed for mobile devices, eliminating the need for specialized hardware and machine learning-based algorithms. The method employs standard cameras and fiducial markers to project realistic shadows on virtual objects, ensuring coherent visual integration with the real environment. The experiments demonstrated that the proposed approach achieved an average angular error of 11.42° for synthetic datasets, effectively estimating scene illumination and generating visually convincing shadow projections. Qualitative tests indicate that the system performs well under various lighting conditions, although it faces limitations in scenarios with translucent objects or diffuse lighting. The results suggest that this solution can serve as an efficient and accessible alternative for augmented reality applications on mobile devices, enhancing immersion and realism without requiring complex computational infrastructure.

Keywords: Augmented Reality, Lighting Estimation, 3D Reconstruction

Edited by: Cleber Gimenez Corrêa | **Received:** 16 June 2025 • **Accepted:** 14 December 2025 • **Published:** 23 January 2026

1 Introduction

Augmented Reality (AR) is described as a technology that integrates virtual and real elements in an interactive and real-time manner [Azuma and Ronald, 1997]. This technology has expanded into various domains, such as gaming, education, industry, and healthcare, offering new ways of interacting with and visualizing information. To ensure convincing AR scenes, it is essential that virtual objects are seamlessly integrated into the real environment, with consistent shadows, geometry, and lighting.

The present study focuses on calculating the directional light vector in AR scenes, enabling accurate shadow projection for virtual objects. This calculation is performed through image processing and inverse rendering techniques, which estimate the light direction in the scene. Instead of relying on deep learning-based approaches, geometric methods are employed, providing a simpler and more straightforward solution. This approach allows for the projection of shadows that are coherent with the environment, enhancing immersion and realism in AR scenarios.

The most common devices that employ AR include Head-Mounted Displays (HMDs), such as AR glasses, and smartphones, which stand out for their versatility and accessibility. Although various devices support this technology, the focus of this work is on the application of AR on mobile phones, given their popularity and ability to provide an interactive experience.

2 Background

With the advancement of mobile devices in recent decades, AR has become more accessible and present in various fields. Nevertheless, achieving proper visual integration between virtual objects and the real environment, particularly regarding lighting and shadow projection, remains a major challenge. As highlighted by Cao and Foroosh [2007], estimating the direction of light in AR scenes using solar shadows is a complex task. Inconsistent lighting undermines both the realism and the immersion of AR experiences.

Previous studies have addressed light estimation in AR using both geometric methods and more advanced techniques. For instance, Cao and Foroosh [2007] and Koc and Balcisoy [2013] computed the direction of light based on shadows and object geometry, whereas deep learning-based methods, such as that proposed by LeGendre *et al*. [2019], offer greater flexibility for complex scenarios.

However, training datasets for shadow generation and manipulation in augmented reality using deep learning methodologies impose substantial costs and complexities. For instance, the recent RdSOBA dataset (Rendered-Shadow-Generation Dataset) released by Tao *et al*. [2024] contains nearly 80,000 object-shadow pairs, 788 foreground 3D objects, 30 rendered 3D scenes, and multiple viewpoints and lighting conditions. Such scale demands high computational resources, time for rendering under multiple lighting setups, annotation or mask generation, and storage, which often limits applicability in low-resource environments.

The present study proposes a geometric approach that does not rely on specialized hardware (e.g., RGB-D cameras, desktop-class GPUs for deep learning) or additional objects (e.g., light probes), using only a standard camera and fiducial markers to compute the directional light vector on an AR scene. This approach optimizes the calculation of shadows and lighting for mobile devices, avoiding the high computational cost and the need for large volumes of data.

2.1 Problem Description

The accurate projection of shadows in AR scenes is essential to ensure convincing immersion and visual integration between virtual objects and the real environment. The central problem lies in precisely calculating the directional light vector so that the illumination of virtual objects corresponds to the lighting conditions of the physical world. However, performing this calculation efficiently, using only a standard camera and fiducial markers, without relying on specialized hardware or complex techniques, remains a significant challenge.

This problem is particularly important in AR scenarios applied to mobile devices, where processing and resource limitations make it even more challenging to ensure realistic and coherent shadow projection within the environment. As highlighted by Kán and Kafumann [2019], lighting estimation on mobile devices must be both accurate and efficient so that virtual objects visually behave naturally, without appearing out of place in the environment. Furthermore, variations in lighting conditions, such as direct or diffuse light, add an extra layer of complexity to the precise calculation of the light vector, requiring a robust solution that performs well across different scenarios.

The choice of a geometric approach and the use of simple smartphone cameras and fiducial markers, instead of more complex techniques such as deep learning or specialized hardware, is justified by the need to develop an accessible, efficient solution suitable for mobile devices. In AR scenarios, where processing and resource limitations are significant, methods that require high computational power or large volumes of training data may not be feasible.

Deep learning techniques, such as those proposed by LeGendre *et al*. [2019], although effective in more complex scenarios, present practical challenges, including the requirement for specialized hardware and high computational costs, which may limit their application on mobile devices. The approach presented in this work, on the other hand, is based on geometric methods that utilize information directly extracted from the scene through a single camera and fiducial markers, such as those provided by ArToolKit [Kato and Billinghurst, 1999]. This enables a low-cost solution applicable in contexts where the use of advanced technologies is unfeasible.

Moreover, the proposed approach aims to maintain efficiency without sacrificing the quality of lighting and shadow estimations, which are essential for a seamless integration between virtual objects and the real environment. This choice is particularly relevant for applications targeting mobile devices, where real-time performance and low resource consumption are critical. The justification for developing this solution lies in the pursuit of an accessible, efficient, and practical tool to project accurate shadows in AR scenes without the need for significant investments in technological infrastructure.

2.2 Materials and Methods

The method proposed in this work to estimate the directional light vector in AR scenes follows a pipeline of interconnected steps, utilizing simple and accessible resources such as mobile phone cameras and fiducial markers. Mobile device cameras enable real-time scene capture without the need for specialized equipment, such as depth cameras or multiple cameras.

Fiducial markers are widely used in AR systems to determine the position and orientation of virtual objects relative to the real environment. These markers, such as those provided by the *ArToolKit* tool developed by Kato and Billinghurst [1999], serve as reference points in the captured image, facilitating the correlation between the real and virtual worlds. Their application simplifies the projection of virtual objects and, in this context, directly contributes to the calculation of the directional light vector.

The proposed pipeline initially involves capturing a 2D image of the scene using a mobile phone camera. Next, the objects and shadows present in the image are separated from other elements using image processing techniques. After this separation, the center of mass of the objects and their respective shadows are calculated. With this information, the 2D image data is transposed into the three-dimensional environment through inverse rendering, enabling the calculation of the directional light vector. This process, which relies on limited resources such as a single camera and fiducial markers, provides an efficient and low-cost solution for light estimation on mobile devices.

3 Main Contributions

The primary contribution of this work is the presentation of an efficient methodology for estimating the directional light vector in AR scenes, using only a mobile device camera and fiducial markers. This approach ensures accurate shadow projection on virtual objects without the need for specialized hardware or complex algorithms such as those based on deep learning.

Specifically, the aim is to implement an image processing pipeline capable of separating objects and shadows from 2D images captured by simple cameras. From this information, inverse rendering techniques are applied to transpose the 2D image data into the three-dimensional environment, enabling the calculation of the light vector. The work also seeks to validate the methodology in different AR scenarios, evaluating the system's efficiency and accuracy.

4 Foundations

This section provides a brief description of the main concepts that will support this work. First, AR is discussed, addressing its main characteristics. Next, the inverse rendering technique is explored, with emphasis on methods such as Differentiable Rendering (DR) and Inverse Ray Tracing (IRT), which are used both in related works and in the present study. Finally, image processing is covered, focusing on segmentation methods such as K-means clustering and GrabCut, as well as techniques that assist in manipulating visual information, which are fundamental to the success of the applications developed throughout this work.

4.1 Augmented Reality

AR is a technology that combines elements from the real world and the virtual world, providing an interactive and immersive experience. A clearer and more modern definition of AR was presented in [Azuma and Ronald, 1997]. The work highlighted three main characteristics: the combination of real and virtual objects in the same environment; interaction

with these objects in real time; and the correct alignment of virtual objects in a three-dimensional space. These characteristics are essential for AR systems to function convincingly, making virtual objects naturally integrate into the physical environment and providing a more coherent experience for the user.

An important aspect of augmented reality is the perception of lighting and shadows, which plays a crucial role in visual realism and user immersion. The projection of realistic shadows is essential for virtual objects to appear truly integrated into the physical environment, helping to provide visual cues about depth and position. In Kán and Kafumann [2019], it was emphasized that, on mobile devices, calculating shadows in real time while considering natural lighting conditions enables virtual objects to visually behave similarly to real objects.

4.2 Inverse Rendering

Inverse rendering can be described as "the problem of estimating one or more properties of lighting, reflectance, and shape from the observed appearance (i.e., one or more images)" [Yu and Smith, 2019]. It is a technique that reverses the traditional rendering process, using 2D images to deduce the physical properties of a 3D scene. This approach is particularly relevant in areas such as AR, where the integration of virtual objects into real environments depends on an accurate estimation of lighting conditions and material characteristics.

In recent years, neural networks have been employed in inverse rendering to estimate three-dimensional information from two-dimensional images [Kato *et al*., 2020]. However, training these networks requires large volumes of 3D data, which are more difficult to obtain compared to 2D images. To overcome this limitation, some recent approaches use 2D data as a reference, adjusting training to improve 3D estimation. One strategy for this is to integrate the inverse rendering process into the neural network pipeline, allowing the generated results to be directly compared with real images and refined based on the observed difference.

Differentiable Rendering (DR), as described by Kato *et al*. [2020], involves a series of techniques aimed at optimizing the rendering process, enabling the system to obtain useful gradients from the rendering procedure. This allows the neural network to efficiently adjust 3D entities even when working with 2D inputs. By integrating the rendering process into network training, more precise and faster estimates of scene geometry and lighting can be achieved.

According to Azinovic *et al*. [2019], when the 3D geometry of the scene is already known, other techniques such as Inverse Ray Tracing (IRT) or Inverse Path Tracing (IPT) may be more suitable for obtaining information about scene illumination. These techniques allow better estimation of light interaction with the already defined scene geometry, especially in AR environments.

Inverse Ray Tracing (IRT), in particular, is a technique that traces rays of light in the scene back to the point of observation. In the context of AR scenes, IRT can be used to calculate the distance between 3D points and the camera by tracing shadows observed in 2D images. When the shadow of an object is captured in a 2D image, IRT allows mapping this shadow back into three-dimensional space, using the shadow's center position to perform ray casting and determine its projection onto the 3D plane. This ensures that virtual shadows are consistent with the real lighting of the scene, creating a more natural and accurate visual integration between virtual objects and the physical environment.

4.3 Image Segmentation

Image segmentation consists of dividing an image into homogeneous regions with the objective of isolating areas of interest, such as shadows and objects, enabling the extraction of relevant information for analysis and processing. For this task, the present work is based on three main techniques: the K-means algorithm, the GrabCut method, and thresholding.

K-means clustering, as presented in Macqueen [1967], is a clustering algorithm that partitions data into k groups or clusters. In the context of segmentation, each image pixel is represented by its features, such as color or intensity values, and is assigned to the cluster whose centroid, calculated based on a distance metric (usually Euclidean), is the closest. The process begins with the random selection of k centroids, after which the pixels are assigned to the corresponding clusters. Then, the centroids are recalculated as the mean of the pixels in each cluster, and this iteration repeats until the assignments stabilize. An important peculiarity of K-means is its sensitivity to the initial choice of centroids, which can lead to varied results for the same image; to mitigate this effect, it is common to use a fixed seed for initialization. In this work, segmentation is performed in the LAB color space, focusing on the L^* component, which ranges from 0 (absolute black) to 100 (absolute white), to identify regions with low luminosity, typically associated with shadows.

Another widely used method is GrabCut, developed by Rother *et al*. [2004]. It is an iterative segmentation technique that extends graph cut approaches for extracting objects from an image. Initially, it is necessary to define a rectangle that bounds the area where the object of interest is presumed to be; this rectangle serves as a starting point for the algorithm to identify, based on similarity in color and texture, which pixels belong to the object and which belong to the background. From this definition, the algorithm constructs a graph where each pixel is represented by a node, and the connections between pixels are weighted according to their similarity in color and texture. To model the color distributions of the object and background, a Gaussian Mixture Model (GMM) is used, characterized by statistical parameters such as the mean (μ) and the standard deviation (σ). GrabCut minimizes an energy function that combines a data term (evaluating the likelihood of pixels fitting the color models) and a smoothness term (enforcing consistency among neighboring pixels). This minimization, performed via max-flow/min-cut algorithms, results in a binary mask that separates the object from the background. The process is iterative, with continuous updates of the GMMs and reassessment of the segmentation, progressively refining the object contours.

Thresholding is a straightforward segmentation approach used when there is a clear distinction between the intensity levels of the objects and the background. In this method, a threshold value is defined, which can be determined automatically, as in Otsu's method Otsu [1979], so that pixels with intensity above the threshold are assigned to one class (e.g.,

white, with value 255) and those below to another class (e.g., black, with value 0). The objective is to maximize the separation between classes by minimizing intra-class variance or, equivalently, maximizing inter-class variance. This conversion produces a binary image that facilitates the creation of masks to isolate regions of interest, although its effectiveness depends on uniform lighting and a clear distinction between the object and the background.

5 Related Work

Geometric methods have been widely used to estimate ambient lighting by analyzing the interaction of light with scene elements to infer its direction and intensity [Koc and Balcisoy, 2013]. Historically, research in this field began primarily with image processing techniques and geometric methods, which analyze the shape of objects and the influence of lighting in the environment. These methods stand out for their simplicity and efficiency but face limitations in dynamic scenarios or those with multiple light sources.

Some works, for example [Castro *et al*., 2012], added physical objects to the scene, including light probes, and used images captured from these objects to extract the light source direction. Our method differs by relying on shadows cast by real objects, without the need to include additional elements.

In recent years, with the advancement of deep learning, techniques have emerged that are more robust in estimating lighting in complex environments. However, these approaches still present challenges, such as the need for large volumes of training data and difficulty handling varying lighting conditions, making light estimation an ill-posed problem, that is, lacking a unique solution for each scene [Marques *et al*., 2022].

Since the proposal of this work does not involve deep learning techniques, this section will focus exclusively on studies that employ geometric methods for lighting estimation. It will present research that explores the relationship between light and shadows in the scene, using image processing techniques and geometric analysis to infer the direction of illumination.

5.1 Light Source Estimation Using Geometry

Light source estimation based on geometry is widely used in AR, visual effects, and computer vision applications. These methods rely on analyzing the geometry of objects present in the scene and the shadows they cast to calculate the direction and intensity of the incident light. One advantage of this type of approach is that it depends on simple physical properties, such as the shape of objects and shadow correspondence, eliminating the need for complex models or large training datasets. This section reviews works that employ geometric techniques for light estimation, focusing on AR environments.

The technique developed by Wang and Samaras [2003] utilizes multiple directional light sources from a single image to estimate the lighting of objects with known geometry and Lambertian reflectance. They combined shading information and cast shadows to determine illumination in AR scenes without requiring calibration objects. Compared to other methods, this approach yielded better results when working with multiple light sources. However, it requires known geometry, which may limit its applicability in some scenarios.

Camera calibration and the estimation of light source orientation using solar shadows were the focus of the work by Cao and Foroosh [2007]. In this study, a method was developed to calculate parameters such as focal length, aspect ratio, and principal point of the camera. The method was capable of estimating the direction of sunlight from two viewpoints of a scene, demonstrating accuracy in both synthetic and real images. The method stood out by dispensing with complex calibration objects, being applicable in natural environments. However, its main limitation lies in the dependence on sunlight, which makes it unsuitable for indoor scenarios or those with artificial light sources.

In the field of light estimation from a single image, Nguyen and Le [2012] proposed a method based on the use of convex-shaped objects as light probes. The system calculates light directions and intensities based on contours defined by the user, eliminating the need for physical light probes and prior knowledge of the scene's 3D geometry. This method demonstrated good accuracy in estimating zenith angles but encountered difficulties when dealing with reflective or transparent surfaces.

Real-time lighting estimation was also explored by Gruber *et al*. [2012], who developed a system based on arbitrary geometry captured by RGB-D cameras. The method uses spherical harmonics to calculate ambient lighting and render soft shadows on virtual objects, ensuring coherent visual integration. The system proved efficient in handling dynamic changes in light sources and the scene but is limited to diffuse Lambertian surfaces, which may restrict its use with reflective materials.

In outdoor AR scenes, ambient lighting estimation can be performed through Lambertian surfaces, as proposed by Koc and Balcisoy [2013]. This work used human face geometry to extract the direction and intensity of ambient light, realistically illuminating virtual objects. The system demonstrated effectiveness on mobile devices, especially under direct sunlight, but showed limitations in cloudy scenarios, where diffuse light affects estimation accuracy. Another positive aspect was its integration with common sensors in mobile devices, allowing the system to operate in real time.

The use of optimization and albedo removal to estimate multiple light sources was proposed by Lopez-Moreno *et al*. [2013], who developed a system capable of preventing the merging of nearby lights. The method was successfully applied in complex scenarios, demonstrating significant accuracy in estimating light directions and intensities, with an average error of 20-30 degrees. However, the technique relies on the manual selection of convex objects, which limits its applicability in scenes with irregular geometric shapes.

Matching the edges of shadows with the surfaces that cast them is a technique that does not require prior knowledge of the object's geometry, as demonstrated by Chotikakamthorn [2015]. The method uses RGB-D depth images to estimate the location of nearby point light sources. Results indicated that the method is effective in estimating light direction in indoor environments but showed greater distance error in scenes with smaller objects.

The use of RGB-D cameras, such as the Kinect, was

also explored by Boom *et al*. [2017], who demonstrated a hybrid CPU-GPU method to efficiently estimate the position of a point light source. The system relies on depth maps and intensity images to perform lighting calculations, enabling realistic rendering of synthetic objects in AR scenarios. With an average accuracy of 20 degrees, the system proved effective, although it has limitations when multiple light sources are present in the scene.

Another method for estimating light direction in AR scenes, using shadows and foreground objects from a single image, was proposed by Liu and Wu [2022]. This technique stands out by employing a homography transformation to align the image coordinate system with the real-world coordinate system, increasing the accuracy of the lighting estimation. Tests conducted in virtual scenes showed low average errors, depending on the scene, with good results in cases with clear shadows. The system also integrates direct manual interaction, allowing users to touch virtual objects more naturally. However, accuracy decreases in scenes where shadows are partially obstructed or where the object is not fully connected to the ground, and the approach is limited to a single directional light source.

Table 1. Comparison between related works

Work	**Inputs**	**Outputs**
Cao and Foroosh [2007]	Two 2D images	3D light vector
Gruber *et al*. [2012]	RGB-D images	Spherical harmonics lighting
Boom *et al*. [2017]	RGB-D images	3D light vector
Chotikakamthorn [2015]	RGB-D images	Multiple 3D light vectors
Wang and Samaras [2003]	2D image	Multiple 3D light vectors
Nguyen and Le [2012]	2D image	Multiple 3D light vectors
Lopez-Moreno *et al*. [2013]	2D image	Multiple 3D light vectors
Koc and Balcisoy [2013]	2D image	3D light vector
Liu and Wu [2022]	2D image	3D light vector
Ours	2D image	3D light vector

Table 1 provides a comparison of different approaches for obtaining scene illumination, based on the types of inputs and outputs required by each method. Geometric methods for illumination estimation explore various strategies to recover information about light from images. Some methods use depth sensors, such as RGB-D cameras, to reconstruct the three-dimensional position of the light source, leveraging both depth and color information simultaneously [Boom *et al*., 2017; Chotikakamthorn, 2015; Gruber *et al*., 2012]. Other approaches operate with 2D images, analyzing the relationship between cast shadows and objects in the scene to estimate the illumination direction [Cao and Foroosh, 2007; Koc and Balcisoy, 2013; Liu and Wu, 2022]. Additionally, some techniques use predefined geometric models, such as Lambertian surfaces or convex objects, to infer multiple light directions [Wang and Samaras, 2003; Nguyen and Le, 2012; Lopez-Moreno *et al*., 2013]. Each of these approaches presents distinct advantages and challenges, being more suitable for different types of scenes and applications, ranging from controlled indoor environments to open spaces with natural lighting.

The work presented in Liu and Wu [2022] is the most similar to our proposal. Aside from the techniques used to segment the image and estimate the light direction, the main difference between our approach and that of Liu and Wu [2022] lies in the use of additional mobile device sensors, such as accelerometers, to calculate the light direction. In contrast, our method relies solely on a single image for estimating the light direction.

6 Proposed Method

This section discusses the AShE system, developed to estimate the directional light vector in AR scenes, with a focus on mobile devices. Although it runs as a web application, all processing is performed locally on the device; no data is sent to the cloud. The user captures an image of the real scene, which is then processed on the device by a pipeline that combines image segmentation and geometry techniques to compute the scene's light vector.

For the system to operate correctly, the captured image must contain some essential elements. It is necessary to have a fiducial marker placed on a flat surface, one or more objects near the marker that cast hard shadows, and a predominant light source, preferably sunlight. The light source should be sufficiently distant to ensure that the shadows of multiple objects remain approximately parallel. These components allow the segmentation algorithms to isolate the objects and their shadows, which are fundamental for the calculation of the directional light vector.

On the other hand, certain conditions can compromise the system's performance. Diffuse or weak lighting makes it difficult to distinguish between the object and the shadow. Translucent objects, those with very dark colors, or varied textures may be mistakenly interpreted as shadows by the algorithms. Additionally, shadows overlapping the objects themselves or other elements of the scene also pose a problem for segmentation.

From this image, the processing pipeline performs segmentation, identifies the centers of mass of the object and the shadow, and calculates the light direction using geometric and inverse rendering techniques. The details of this process, as well as the overall system architecture, are presented in the following sections.

6.1 System Architecture

The system was developed using accessible and widely used technologies, aiming to create a solution that could run directly in browsers without the need for external servers. This approach was chosen to ensure system accessibility, especially on mobile devices.

The segmentation algorithms used do not depend on high-quality images. Therefore, it is possible to use the system on devices with simple cameras, such as cell phones and webcams. This aligns with our initial goal of accessibility.

For image processing, the OpenCV library was used, known for its widespread application in the field of computer vision. This library enabled the use of algorithms such as GrabCut and K-means to segment the image, isolating objects and their shadows. All processing is performed directly in the browser through the Pyodide library, which allows Python

code to run within the JavaScript environment.

The web interface was built using HTML, CSS, and JavaScript, while the Three.js and AR.js libraries were employed to render 3D objects and integrate the virtual scene with the real environment. The combination of these tools enabled the creation of AR scenes directly within the web environment. The source code and datasets of our system is available in https://github.com/AVRGroup/AShE.

The developed system is structured around the processing pipeline shown in Figure 1 and each step is illustrated in Figure 2.

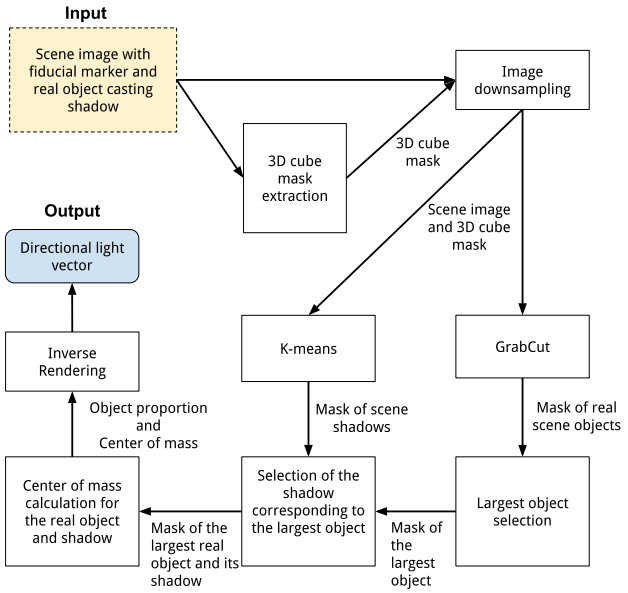
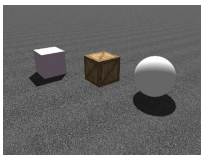


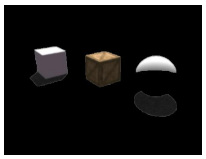
Figure 1. AShE's processing pipeline.

Each step plays a fundamental role in the overall functioning of the system. Therefore, each of these steps will be detailed throughout this section, highlighting the methods employed.



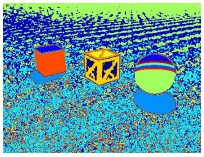
(a) Input - Scene image

(b) Input - Cube mask



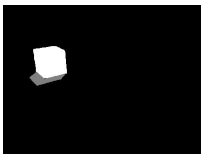
(c) Segmentation - GrabCut

(d) Segmentation - GrabCut mask

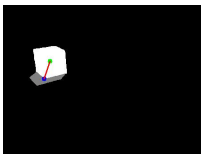


(e) Segmentation - K-means application

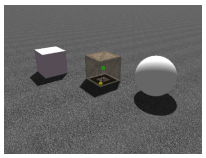
(f) Segmentation - K-means mask



(g) Largest object and its corresponding shadow



(h) Centers of mass - 2D vector



(i) Inverse Rendering - 3D estimation

Figure 2. System pipeline visualization.

The system input consists of an image containing essential elements, as previously described in this work. Subsequently, geometric methods are applied to associate shadows with their corresponding objects. Finally, using inverse rendering techniques, the system calculates the directional light vector, enabling the projection of virtual shadows aligned with the scene's lighting conditions.

Each of these steps will be detailed in the following subsections, covering everything from data input to geometric processing and final rendering.

6.2 Data Input

The system's data input consists of two distinct images that together provide the necessary information for processing. The first image is a capture of the real scene, obtained through a standard camera (Figure 2(a)). This image must contain the fiducial marker, nearby real objects that cast shadows, and the predominant light source (preferably sunlight). The second image is generated by rendering the virtual object using *AR.js*, which displays the cube with a single color, enabling precise extraction of its position and boundaries, resulting in a mask (Figure 2(b)). The cube's mask is obtained from an intermediate rendering performed on a *render target*, where the cube is highlighted with a uniform color material. This mask allows isolating the virtual object from the scene.

To optimize performance, the captured scene image is resized to a resolution of 640 × 480 pixels, balancing the quality required for analysis with reduced memory usage.

6.3 Image Segmentation

Two main algorithms were used: K-means, employed for shadow detection; and GrabCut, responsible for segmenting the foreground objects.

The first step in segmentation involves detecting shadows using K-means. This algorithm groups the image pixels into different clusters based on their color and brightness characteristics, as illustrated in Figure 2(e). In the LAB color space, the darkest clusters are classified as shadows. As a result, binary masks are generated where shadow pixels are marked in white (255) and all other pixels in black (0), as shown in Figure 2(f). To ensure reproducibility across different runs, a fixed seed was set, preventing unwanted variations in the segmentation results. The parameters of the method were empirically defined through experimentation.

The segmentation of foreground objects is performed using GrabCut. Initially, a region of interest is defined as a rectangle that covers almost the entire image. This strategy enables automation of the process without requiring manual intervention. From this initial region, the segmentation is refined iteratively, separating the foreground from the background, as shown in Figure 2(c). The final output produced by GrabCut is a binary mask where the segmented objects appear isolated, as illustrated in Figure 2(d).

6.4 Association Between Object and Shadow

After image segmentation, the system generates two sets of binary masks representing objects (Figure 2(c)) and shadows (Figure 2(f)). To prevent the virtual 3D object from being considered in the analysis, its mask (Figure 2(b)) is applied to

exclude it from the subsequent processing steps. Among the object masks, the one with the largest area is selected, since the GrabCut algorithm may produce small unwanted artifacts. This criterion ensures that the selected mask corresponds to a real object, reducing noise interference.

After selecting the largest region corresponding to an object, its center of mass is calculated, defined as the average position of all pixels that make up the identified region. This calculation is performed using the following formulas:

$$cx = \frac{M_{10}}{M_{00}}, \quad cy = \frac{M_{01}}{M_{00}}, \tag{1}$$

where cx and cy correspond to the horizontal and vertical coordinates of the center of mass, M_{00} is the total area of the region, represented by the sum of the *pixels* belonging to the object, and M_{10} and M_{01} are the spatial moments related to the distribution of the pixels with respect to the horizontal and vertical axes.

This calculation is applied both to the largest object and to each identified shadow. From this, the system uses criteria based on distance and size to determine which shadow belongs to the main object. The Euclidean distance is calculated between the object's center of mass and the centers of mass of each shadow. This criterion ensures that shadows closer to the object are prioritized.

Additionally, the system considers the relative area of the shadows, penalizing those that are small or far from the object. This penalization is implemented through an exponential function, adjusting the area according to the distance:

$$j = \frac{a}{e^{d/c} + k}, \tag{2}$$

where j is the adjusted area, a is the area, d is the distance, c is a scaling constant that controls the impact of the distance, and k prevents division by zero. Based on the distance and adjusted area, each shadow receives a score s calculated as:

$$s = \frac{d}{j + 1}. \tag{3}$$

The shadow with the lowest score is selected as corresponding to the main object. This association method is effective in scenarios with hard shadows and clearly visible objects. However, it may present limitations in situations where shadows overlap or are cast on irregular surfaces. The result of associating the largest object with its shadow is shown in Figure 2(g) of the pipeline.

6.5 Processing the Directional Light Vector

The calculation of the directional light vector is the final stage of the proposed pipeline, employing inverse rendering techniques to determine the direction of light based on elements extracted from the two-dimensional image. This step combines geometric information from the scene with specific adjustments to correlate the processed data from the 2D image to the three-dimensional space.

To perform this calculation, the system uses the center of mass of the largest real object to correspond to the central point of the 3D model generated on the fiducial marker, establishing it as a reference in three-dimensional space. Since the system does not directly have the shadow of the 3D model, it assumes that if the real object had the same proportion and size as the virtual object, their shadows would be equivalent. Based on this logic, the system scales the vector between the center of the real object and the center of its shadow (Figure 2(h)) according to the relative proportion between the real and virtual objects. This adjustment allows the estimated position of the virtual model's shadow to be determined, using the characteristics of the real object and its shadow as reference.

After the scale adjustment, the system performs raycasting, which consists of projecting a virtual ray from an origin point (such as the camera position) toward a target, like a specific point on a plane. In the context of the proposed system, this technique is used to project the 2D point corresponding to the center of mass of the real object's shadow onto the three-dimensional plane of the virtual scene. The vector calculated between the center of the real object and the center of its shadow is scaled according to the previously computed adjustment, ensuring that the raycasting intersection point provides an appropriate estimate of the virtual model's shadow position in 3D coordinates.

With the three-dimensional points of the geometric center of the virtual model and the estimated position of the projected shadow determined, the directional light vector is calculated as the vector difference between these two coordinates in three-dimensional space. This vector represents the direction of the incident light in the scene, which is then applied to the virtual scene's lighting, enabling the generation of shadows consistent with the real environment's illumination. Figure 2(i) visually represents the generated vector between the central point of the virtual object (green point) and the raycasting intersection point with the plane (yellow point).

Additionally, the system allows the user to switch the virtual models associated with the fiducial marker after the light vector has been processed. This enables real-time visualization of how the shadows of different virtual objects adapt to the calculated lighting.

7 Results

In this section, the results obtained by the system are presented. Tests were conducted using both artificial and real images. However, the accuracy of the algorithm could only be determined with the artificial images, since only in these cases is it possible to obtain a reference light angle for the scene.

The real scenarios were captured using a cellphone camera, with a printed fiducial marker fixed to the ground and a real object positioned nearby. The images were taken outdoors between 10 a.m. and 3 p.m., with direct sunlight incidence to produce well-defined hard shadows.

The synthetic scenarios were generated using *Three.js*, with virtual scenes containing an image of the fiducial marker at the center and simple geometric objects around it. This approach allowed manipulation of the directional light vector, enabling the creation of different shadow patterns in the scene. Additionally, textures were added to the ground to simulate surface irregularities.

Performance tests were conducted on two different devices. The first is a mobile device, an iPhone 8 Plus running iOS 16, equipped with an Apple A11 Bionic processor (6

cores: 2 × 2.39 GHz Monsoon + 4 × 1.19 GHz Mistral), a 3-core Apple GPU, and 3 GB of RAM. The second device is a desktop running Windows 10, featuring an AMD Ryzen 7 5700X processor (8 cores, 16 threads, base frequency of 3.4 GHz), an AMD Radeon RX 6750XT GPU, and 32 GB of RAM.

7.1 Artificial Scenarios

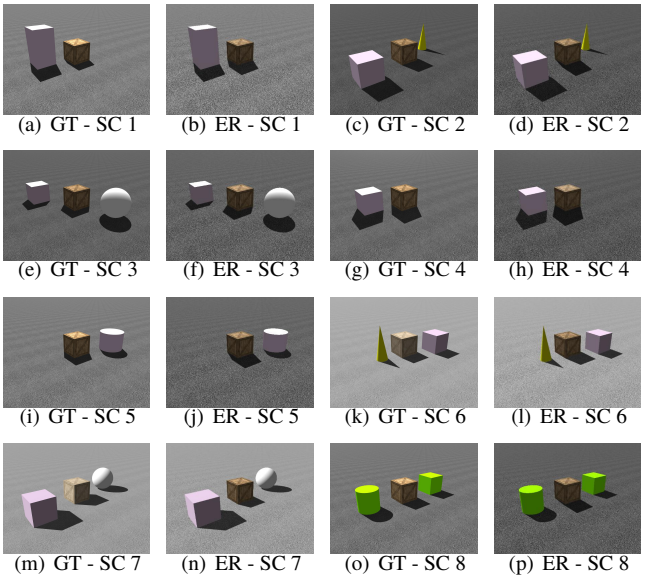
The artificial scenarios allowed the acquisition of the directional light vector for each scene, enabling a direct comparison between the vectors calculated by the system and the actual vectors used to generate the scene. This made it possible to calculate the angular error between these vectors and assess the accuracy of the proposed approach.

When creating the scenes, efforts were made to reproduce ideal conditions for the system's operation. Objects were positioned at an appropriate distance from the marker, avoiding excessive proximity that could compromise shadow definition. Furthermore, objects with light colors and backgrounds with simple, minimally varied patterns were used to reduce interference in the segmentation algorithms.

7.1.1 Quantitative Results

This section presents the quantitative results obtained from the artificial experiments. The main parameter evaluated was the directional light vector estimated by the system, with its accuracy measured by the angular error relative to the ground truth.

Figure 3 shows the results of tests conducted in artificial scenarios. The images are arranged in pairs, with the first representing the ground truth, that is, the directional light vector generated in the virtual scene using *Three.js*, and the second illustrating the light vector estimated by the system. This arrangement allows for a visual comparison between the results.



(a) GT - SC 1 (b) ER - SC 1 (c) GT - SC 2 (d) ER - SC 2
(e) GT - SC 3 (f) ER - SC 3 (g) GT - SC 4 (h) ER - SC 4
(i) GT - SC 5 (j) ER - SC 5 (k) GT - SC 6 (l) ER - SC 6
(m) GT - SC 7 (n) ER - SC 7 (o) GT - SC 8 (p) ER - SC 8

Figure 3. Comparison between Ground Truth (GT) and Estimated Results (ER) in all artificial scenarios (SC).

Table 2 presents the real and estimated vectors for each scenario, as well as the associated angular errors. The average error of 11.42° demonstrates that the system achieves considerable accuracy, especially when compared to the worst possible case, which would be an angular error of 180°. The angular error represents the directional divergence between the vectors, expressed in degrees, indicating the accuracy of the algorithm.

Table 2. Angular error between ground-truth and estimated vectors for eight scenarios.

	Ground-Truth Vector	Estimated Vector	Angular Error (°)
1	$(1.15, 7, -4)$	(-1.15, 1.42, -1.26)	16.12°
2	$(-0.5, 3, -3)$	(0.30, 1.35, -1.44)	15.47°
3	$(4, 8, -4)$	(0.95, 1.52, -0.88)	5.34°
4	$(3, 5, -5)$	(0.95, 1.19, -1.30)	5.80°
5	$(3, 8, -5)$	(0.45, 1.38, -1.37)	13.16°
6	$(-1, 6, -5)$	(-0.25, 1.40, -1.40)	5.15°
7	$(-3, 4, -3)$	(-0.46, 1.63, -1.06)	18.00°
8	$(-1, 6, -5)$	(0.14, 1.42, -1.41)	12.35°
-	**Average**		**11.42°**

A correlation can be observed between the angular errors shown in Table 2 and the image pairs in Figure 3. For instance, in scenario 7 (Figures 3(m) and 3(n)), the estimated shadow of the cube significantly diverges from the real shadow, reflecting the highest angular error found, 18.00°. On the other hand, in scenario 6 (Figures 3(k) and 3(l)), the angular error was the lowest, 5.15°, with the estimated shadow almost perfectly aligned with the real shadow, making the difference barely noticeable.

For the mobile device, the average processing time for the synthetic images was 6.73 seconds, while for the desktop it was 3.86 seconds, representing a 42.6% faster execution on the desktop compared to the mobile. To ensure a more accurate performance evaluation, the initial runs were disregarded since the initial loading of *Pyodide* can significantly affect the processing time. It is important to note that the execution times are one-time; once processing is completed, the image's lighting vector remains fixed, eliminating the need for future reprocessing. After this preprocessing step, which took only a few seconds, the system ran at 60 fps on all tested devices.

7.2 Real-World Scenarios

The images of the real scenarios were captured with the aim of observing the system's performance under uncontrolled conditions. Unlike the artificial scenarios, there is no reference lighting vector for direct comparison, making it impossible to precisely quantify the angular error.

However, these tests served to evaluate the performance of the segmentation algorithms in complex environments. The images made it possible to verify the system's ability to correctly distinguish objects from the background and identify their respective shadows, even in the presence of variations in lighting, object shapes, colors, and background textures.

7.2.1 Qualitative Results

This section presents a qualitative analysis of the results obtained in real-world scenarios. While the previous section provided numerical values of angular errors for artificial scenarios, here the evaluation is based on the visual coherence of the projected shadows, allowing for an assessment of the system's behavior under uncontrolled conditions.

Figure 4 shows the results for the real scenarios. Although it is not possible to calculate the angular error due to the lack of ground truth, visual inspection indicates that the projected shadows maintain good correspondence with the real shadows in the scene. For instance, in Figure 4(a), even with a bright and complex textured background, the shadow of the cube does not deviate noticeably from the shadow of the real cup. Similarly, in Figures 4(b) and 4(c), the darker background does not impair segmentation, allowing the system to produce visually coherent results.

(a) Real scenario 1 (b) Real scenario 2 (c) Real scenario 3

Figure 4. Results in real scenarios with a virtual textured cube.

With the estimated lighting vector, different virtual objects can be integrated into the scene, ensuring that their shadows are projected correctly regardless of their shapes. The initial cube serves only as a reference for calculating the light vector. Once this vector is obtained, the cube can be replaced with other models, allowing the visualization of more complex and realistic objects integrated into the scene, as illustrated in Figure 5.

(a) 1st 3D object (b) 2nd 3D object (c) 3rd 3D object

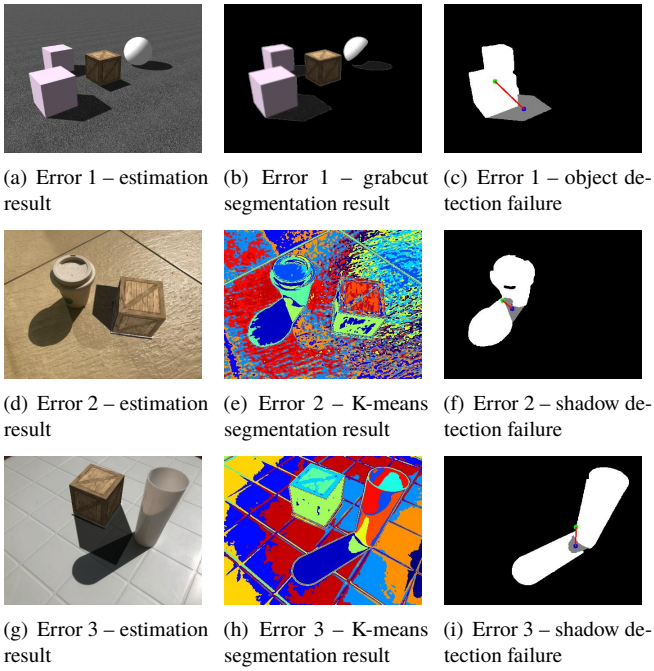
Figure 5. Visualization of scenes with different 3D objects.

Furthermore, it was observed that the system can handle real objects of various shapes and colors, as well as variations in light incidence. Objects with complex shapes, light-colored surfaces, and partially self-cast shadows did not significantly compromise the lighting estimation. The method also demonstrated robustness across different capture angles, distances, and times of day.

8 Limitations

Despite the results presented in the previous sections, the system exhibits some limitations that may impact its accuracy and applicability in different scenarios. During testing, limitations were identified that affect the estimation of the directional light vector, making it necessary to discard some samples that showed processing failures or results significantly deviating from expectations.

One of the main limitations observed was the system's difficulty in handling overlapping or partially occluded shadows and objects. The system assumes that objects and their shadows are visible and isolated, but in cases where multiple objects and shadows merge or are blocked by other elements in the scene, the correspondence between the object and its shadow becomes less precise. This issue can be seen in Figure 6(a), where one cube overlaps another, complicating the segmentation algorithm (Figure 6(b)). This problem directly affects the determination of the light vector (Figure 6(c)), resulting in an imprecise illumination estimate.



(a) Error 1 – estimation result (b) Error 1 – grabcut segmentation result (c) Error 1 – object detection failure

(d) Error 2 – estimation result (e) Error 2 – K-means segmentation result (f) Error 2 – shadow detection failure

(g) Error 3 – estimation result (h) Error 3 – K-means segmentation result (i) Error 3 – shadow detection failure

Figure 6. System errors.

Another significant limitation is the system's sensitivity to diffuse lighting. The method relies on the presence of well-defined hard shadows to accurately estimate the light direction. However, in environments with multiple light sources or diffuse light, such as on cloudy days, shadows may lose contrast or even become imperceptible, making correct segmentation difficult. This issue is illustrated in Figures 6(d), 6(e), and 6(f), where the presence of two light sources results in shadows of varying intensities. The overlapping of these shadows creates a region of higher intensity that is mistakenly interpreted by the algorithm as the main shadow.

Additionally, the system's performance is affected by surfaces with highly varied textures, both in the background and on the objects themselves. In scenarios with complex visual patterns, reflective surfaces, or translucent materials, the segmentation algorithms can confuse parts of the background with shadows or incorrectly separate regions of the main shadow. This problem can be seen in Figures 6(g), 6(h), and 6(i), where the analyzed object is a slightly translucent and reflective glass. Due to its translucency, part of the light passes through the material, generating a shadow with varying intensities along its projection. These variations cause the projected shadow to be inconsistently identified, resulting in the segmentation of the shadow into distinct clusters, which compromises the correct identification of its shape and position. Similarly, detailed patterns in the background can be mistakenly interpreted as objects, interfering with the segmentation process and impacting the illumination estimation.

In terms of performance, the system was designed to run directly in browsers on mobile devices. However, the use of AR elements can be computationally intensive, especially on low-end devices. During testing, it was observed that,

depending on the parameters used in the segmentation algorithms, the complexity of the 3D models, or the quality of the shadows, processing times increased significantly, sometimes causing the application to freeze or even the page to reload.

Finally, it is important to emphasize that the proposed algorithm works correctly only when the shadow of the real object is projected onto a flat surface. Processing shadows projected onto non-planar objects would require implementing additional techniques to infer the geometry of the object receiving the shadow.

These limitations do not invalidate the system's functionality but indicate that there is room for improvement and refinement. Some constraints, such as the dependence on hard shadows, stem from the approach adopted in this work, which is based on image segmentation. Others, such as performance issues, could be addressed by reducing scene quality or by combining additional segmentation techniques.

9 Conclusion

In this work, a system was presented to estimate the directional light vector of a 3D scene from images captured by simple devices, such as mobile phones and webcams, using geometric and segmentation techniques. The main objective was to enable a more realistic integration of virtual objects into real environments, ensuring coherent correspondence between the lighting of synthetic elements and the captured surroundings.

The system pipeline begins with capturing an image containing a fiducial marker, on which a virtual cube is positioned. Subsequently, segmentation techniques are applied to identify objects and shadows in the scene. From this information, the system estimates the directional light vector by analyzing the relationship between the centers of mass of the real object and its shadow. Then, scaling adjustments are performed to correlate the proportions between the real and virtual objects. Finally, inverse rendering techniques are employed to transform the vector obtained from the 2D image into a three-dimensional light vector, which is used to illuminate the scene.

The experiments conducted demonstrated that the system is capable of estimating the light direction with an average angular error of 11.42°. Tests in artificial scenarios indicated that the approach is suitable for controlled conditions. In real scenarios, qualitative evaluation indicated that the system can generate shadows consistent with the scene's actual shadows. However, the absence of a ground truth for numerical validation made quantitative analysis impossible in these cases.

Despite the positive results, some limitations were identified. Image segmentation proved to be the main source of error in the system, showing sensitivity to textured backgrounds, overlapping shadows, and variations in light intensity. Additionally, the presence of multiple light sources or poorly defined shadows can compromise the accuracy of the estimation. Regarding computational performance, the system executed efficiently on most modern devices but may face difficulties on more limited hardware, especially older mobile devices, where segmentation and rendering calculations can impact the scene's frame rate.

Nonetheless, the system demonstrated potential for AR applications in controlled scenarios. The results indicate that geometric approaches can be viable for this type of problem without the need for more complex techniques such as machine learning. As future work, improvements in segmentation, strategies to better handle adverse scenarios, and optimizations may contribute to enhancing the system's quality and applicability.

Declarations

Funding

The authors of this work would like to thank the Minas Gerais Research Funding Foundation (Fapemig) for the financial support provided to carry out this research. Fapemig's support was fundamental to the development of this study, allowing the investigation of new perspectives and contributing to the advancement of knowledge in our research area.

Authors' Contributions

AD was the primary developer of the system and the main author of the manuscript. RLS contributed to the study's conception and provided overall guidance. All authors read and approved the final version of the manuscript.

Competing interests

The authors declare that they have no competing interests.

Availability of data and materials

The source code and datasets generated and/or analyzed during the current study are available in https://github.com/AVRGroup/AShE.

References

Azinovic, D., Li, T.-M., Kaplanyan, A., and Nießner, M. (2019). Inverse path tracing for joint material and lighting estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 2447–2456. DOI: https://doi.org/10.1109/CVPR.2019.00255.

Azuma and Ronald, T. (1997). A survey of augmented reality. *Presence: Teleoperators & Virtual Environments*, 6(4):355–385. DOI: https://doi.org/10.1162/pres.1997.6.4.355.

Boom, B. J., Orts-Escolano, S., Ning, X. X., McDonagh, S., Sandilands, P., and Fisher, R. B. (2017). Interactive light source position estimation for augmented reality with an rgb-d camera. *Computer Animation and Virtual Worlds*, 28(1):e1686. DOI: https://doi.org/10.1002/cav.1686.

Cao, X. and Foroosh, H. (2007). Camera calibration and light source orientation from solar shadows. *Computer Vision and Image Understanding*, 105(1):60–72. DOI: https://doi.org/10.1016/j.cviu.2006.08.003.

Castro, T. K., Figueiredo, L. H., and Velho, L. (2012). Realistic shadows for mobile augmented reality. In *Symposium on Virtual and Augmented Reality*, pages 36–45. DOI: https://doi.org/10.1109/SVR.2012.9.

Chotikakamthorn, N. (2015). Near point light source location estimation from shadow edge correspondence. In *2015 IEEE 7th International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM)*, pages 30–35. IEEE. DOI: https://doi.org/10.1109/iccis.2015.7274543.

Gruber, L., Richter-Trummer, T., and Schmalstieg, D. (2012). Real-time photometric registration from arbitrary geometry. In *2012 IEEE international symposium on mixed and*

augmented reality (ISMAR), pages 119–128. IEEE. DOI: https://doi.org/10.1109/ISMAR.2012.6402548.

Kán, P. and Kafumann, H. (2019). Deeplight: light source estimation for augmented reality using deep learning. *The Visual Computer*, 35(6):873–883. DOI: https://doi.org/10.1007/s00371-019-01666-x.

Kato, H., Beker, D., Morariu, M., Ando, T., Matsuoka, T., Kehl, W., and Gaidon, A. (2020). Differentiable rendering: A survey. *arXiv preprint arXiv:2006.12057*. DOI: https://doi.org/10.48550/arXiv.2006.12057.

Kato, H. and Billinghurst, M. (1999). Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proceedings 2nd IEEE and ACM International Workshop on Augmented Reality (IWAR'99)*, pages 85–94. IEEE. DOI: https://doi.org/10.1109/IWAR.1999.803809.

Koc, E. and Balcisoy, S. (2013). Estimation of environmental lighting from known geometries for mobile augmented reality. In *2013 International Conference on Cyberworlds*, pages 132–139. IEEE. DOI: https://doi.org/10.1109/CW.2013.65.

LeGendre, C., Ma, W.-C., Fyffe, G., Flynn, J., Charbonnel, L., Busch, J., and Debevec, P. (2019). Deeplight: Learning illumination for unconstrained mobile mixed reality. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5918–5928. DOI: https://doi.org/10.1109/CVPR.2019.00607.

Liu, D. S.-M. and Wu, S.-J. (2022). Light direction estimation and hand touchable interaction for augmented reality. *Virtual Reality*, 26(3):1155–1172. DOI: https://doi.org/10.1007/s10055-022-00624-8.

Lopez-Moreno, J., Garces, E., Hadap, S., Reinhard, E., and Gutierrez, D. (2013). Multiple light source estimation in a single image. In *Computer Graphics Forum*, volume 32, pages 170–182. Wiley Online Library. DOI: https://doi.org/10.1111/cgf.12195.

Macqueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability/University of California Press*.

Marques, B. A. D., Clua, E. W. G., Montenegro, A. A., and Vasconcelos, C. N. (2022). Spatially and color consistent environment lighting estimation using deep neural networks for mixed reality. *Computers & Graphics*, 102:257–268. DOI: https://doi.org/10.1016/j.cag.2021.08.007.

Nguyen, R. M. and Le, M. N. (2012). Light source estimation from a single image. In *2012 12th International Conference on Control Automation Robotics & Vision (ICARCV)*, pages 1358–1363. IEEE. DOI: https://doi.org/10.1109/ICARCV.2012.6485343.

Otsu, N. (1979). A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62–66. DOI: https://doi.org/10.1109/TSMC.1979.4310076.

Rother, C., Kolmogorov, V., and Blake, A. (2004). Grabcut: interactive foreground extraction using iterated graph cuts. *ACM transactions on graphics (TOG)*, 23(3):309–314. DOI: https://doi.org/10.1145/1015706.1015720.

Tao, X., Cao, J., Hong, Y., and Niu, L. (2024). Shadow generation with decomposed mask prediction and attentive shadow filling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 5198–5206. DOI: https://doi.org/10.1609/aaai.v38i6.28326.

Wang, Y. and Samaras, D. (2003). Estimation of multiple directional light sources for synthesis of augmented reality images. *Graphical Models*, 65(4):185–205. DOI: https://doi.org/10.1016/S1524-0703(03)00043-2.

Yu, Y. and Smith, W. A. (2019). Inverserendernet: Learning single image inverse rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3155–3164. DOI: https://doi.org/10.48550/arXiv.1811.12328.